Traffic Sign Recognition Classifier 17.08.2017

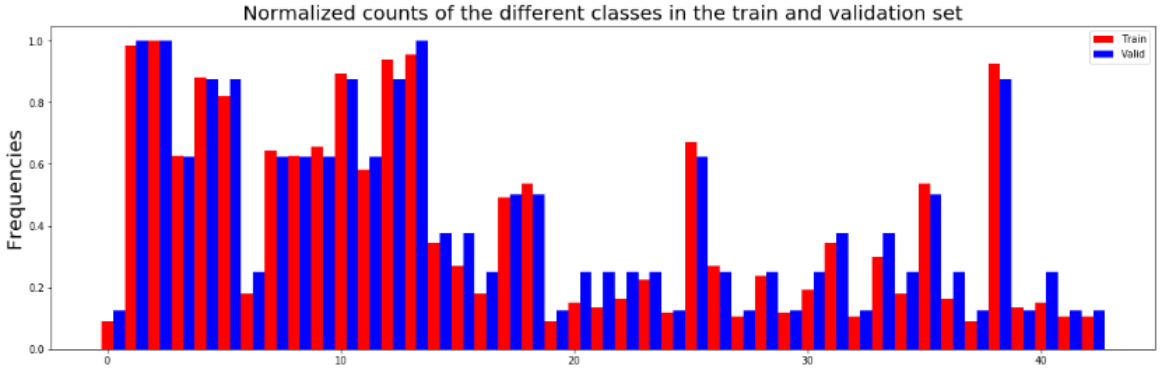
This is the report of my submission to the ''Project: Build a Traffic Sign Recognition Classifier'' for Term 1 of the Self-Driving Car Engineer Nanodegree of Udactiy.

Data Set Summary & Exploration

I used numpy to count the number of training, validation, and testing examples and the number of classes.

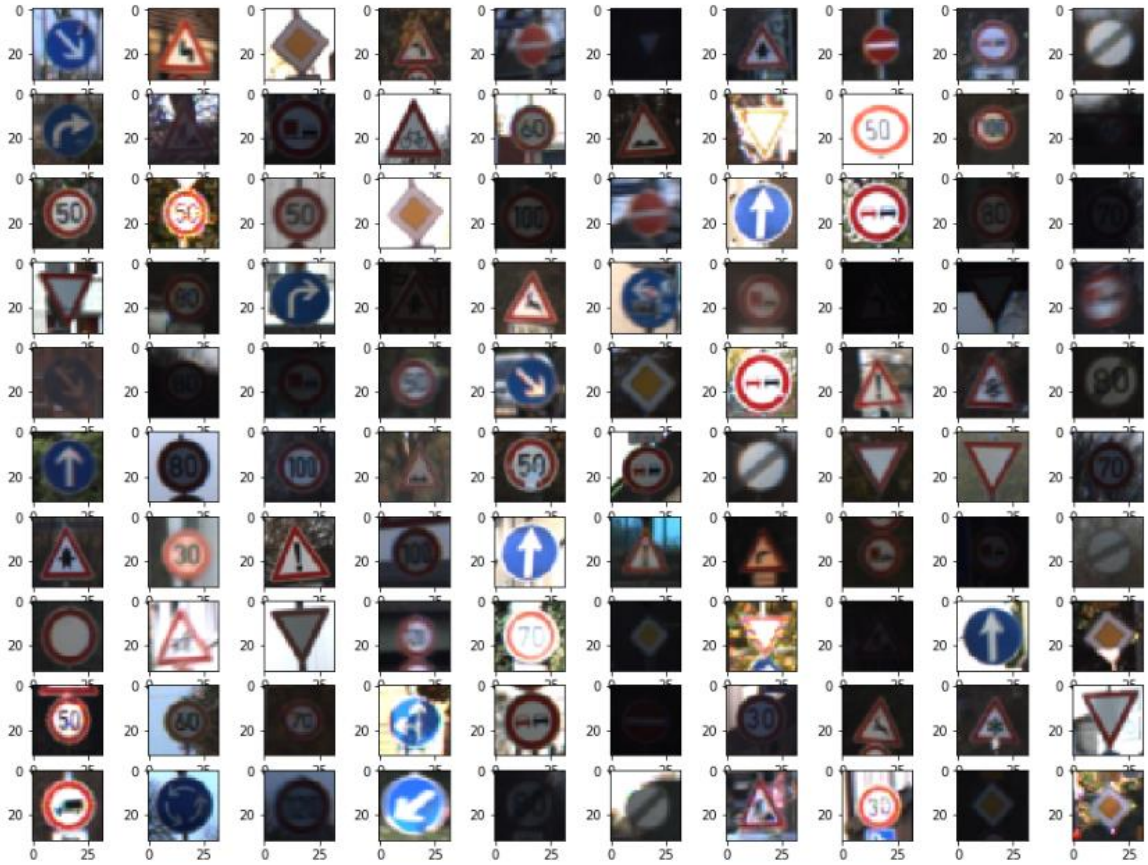
Size of the training set:	34799
Size of the validation set:	4410
Size of the test set:	12630
Number of classes:	43

Further, I counted for each class the number of examples per class in the training and validation set. I normalized the frequencies of these counts. The resulting bar plot looks as follows.



First, I observed that the frequencies of the different classes vary in both sets. However, the distribution is rather similar in both sets. Hence, we conclude that the validation set captures important aspects of the training set.

Further, I plotted 100 random training images to get a better intuition of the dataset.



Design and Test a Model Architecture

I tested, three different preprocessing procedures: Conversion to gray scale and centering the data set, subtracting the average image from all images and centering each image channel from the data set.

I tested different network structures: A variation of the VGG Net, LeNet presented in the course with SAME and VALID Padding, with and without dropout layers.

My final model is similar to the LeNet model presented in the lecture. I added two dropout layers after each RELU nonlinearity in the fully connected layers to provide overfitting as well as L2 regularization.

Model architecture

Layer	Description
Input	32x32x3 RGB image
Convolution 5x5	1x1 stride, valid padding, Output 28x28x6
RELU	Output 28x28x6
MAX pooling 2x2	2x2 stride, valid padding, Output 14x14x6
Convolution 5x5	1x1 stride valid padding, Output 10x10x6
RELU	Output 10x10x6
MAX pooling 2x2	Output 5x5x6
Flatten	Output 400
Fully connected	Output 120
RELU	Output 120
Dropout	Output 120
Fully connected	Output 84
RELU	Output 84
Dropout	Output 84
RELU	Output 84
Fully connected	Output 10
Softmax	Output 10

Initialization

To initialization the model I set all bias terms to 0 and sampled the weights from a normal distribution with mean 0 and standard deviation 0.1.

Training the model

The following hyperparameters were used to train the model:

Epochs:	30
Batchsize:	128
Learningrate:	0.001
Regularization scaling:	0.03
Dropout probability:	0.5

To updates the weights I used the Adam update rule. I saved the model with the best validation accuracy. The training was performed on my GTX 670 and took about 165 seconds.

Finding the architecture

The first models, I tested were to powerful, they achieved training accuracies of almost 100% but the validation accuracy lacked behind. Therefore, I decided to add dropout and L2 regularization. Tuning the scale of L2 regularization, I could achieve better performance on the validation set sacrificing little performance on the training set.

Performance of the model

My final model results were:

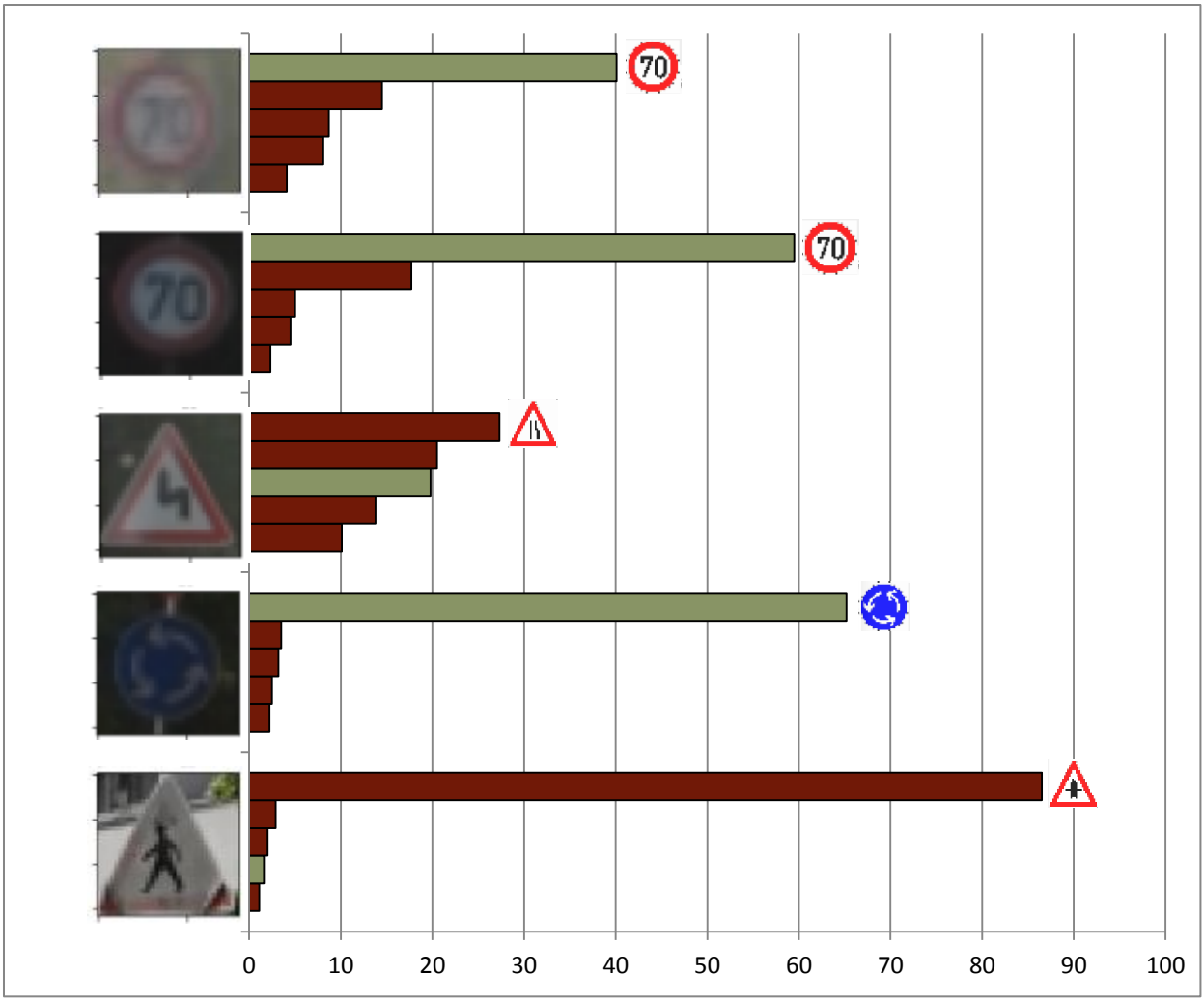
Training set accuracy:	0.982
Validation set accuracy:	0.961
Test set accuracy:	0.938

Test a Model on New Images

Living in Germany, I was able to use my cell phone to take photos of traffic signs. After taking the pictures, I selected manually the parts of the image which contained the traffic sign. The following 18 photos are tested with my model.

I expected the image in the third row and fourth column to be difficult to classify since the border of the traffic sign is really bleached.

My model predicted 16 out of these 18 images correctly, which results in an accuracy of 89%. The model is rather certain about the classification of most images. However, for the following 5 images the model is relatively uncertain or even wrong.



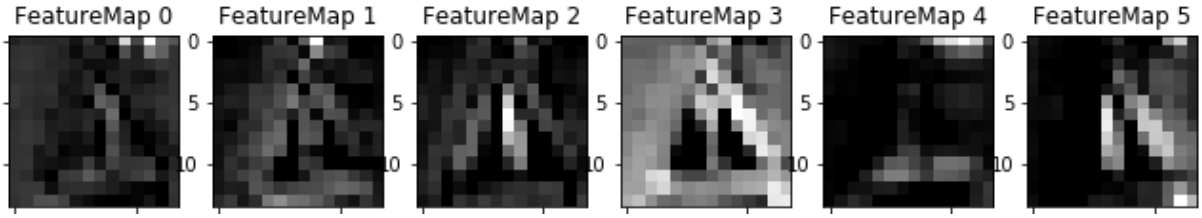
The predicted probabilities for five different images of my test set. The green bars show correct predictions the red bar show wrong predictions.

As expected the model has problems identifying the traffic sign with the bleached border. It focuses on the black part in the middle and thinks it is a “right-of-way at the next intersection” sign. The second error of the model is also plausible since the “road narrows on the right’’ sign looks very similar to the ‘’double curve’’ sign.

Visualizing the Neural Network

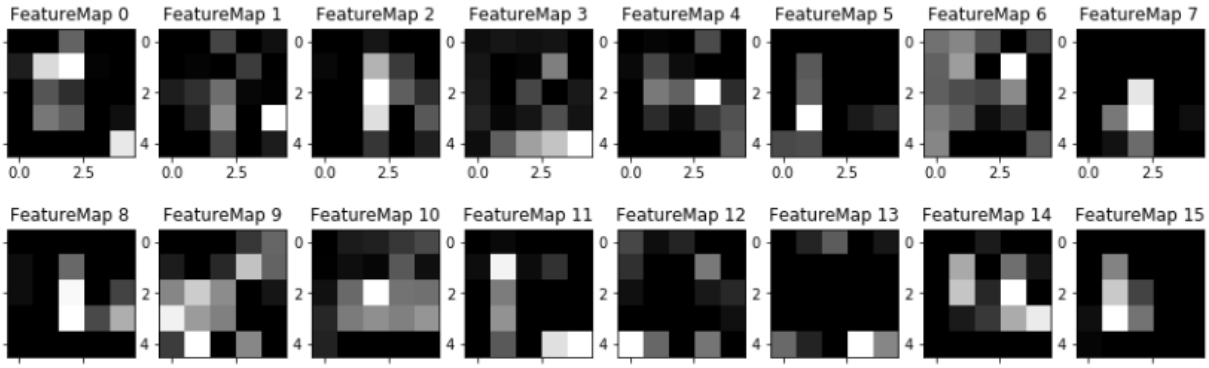
I used the first image of my self-defined test set as a stimulus for the network.

My network shows the following activations on the first layer



As can be seen, some feature maps have learned to search for specially oriented edges

My network shows the following activations on the second layer



The second layer combines the features defined by the first layer. The actual meaning of these feature maps is impossible to interpret with a single stimulus.

Improvement of the Model

The model is able to fit the training set very well but shows worse performance on the validation and test set. To improve the general performance of the model I plan to use data augmentation to increase the size of the training set. Especially the examples which correspond to classes which are less frequent in the training set should be augmented.